

Research software review as part of the publication process

SciCodes - Consortium of scientific software registries and repositories
October 21, 2021

Ana Trisovic, Harvard University

Why review research code?

- Necessary for research verification
 - Critical for reproducibility
 - It can enhance transparency of research
 - Facilitates reuse and building upon previous results
-

Presentation agenda



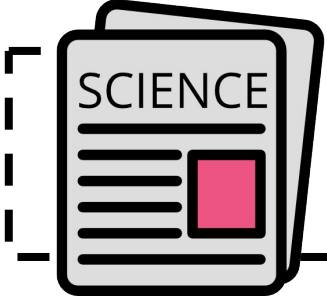
Researchers

Research data
and code



Research data or
software repositories

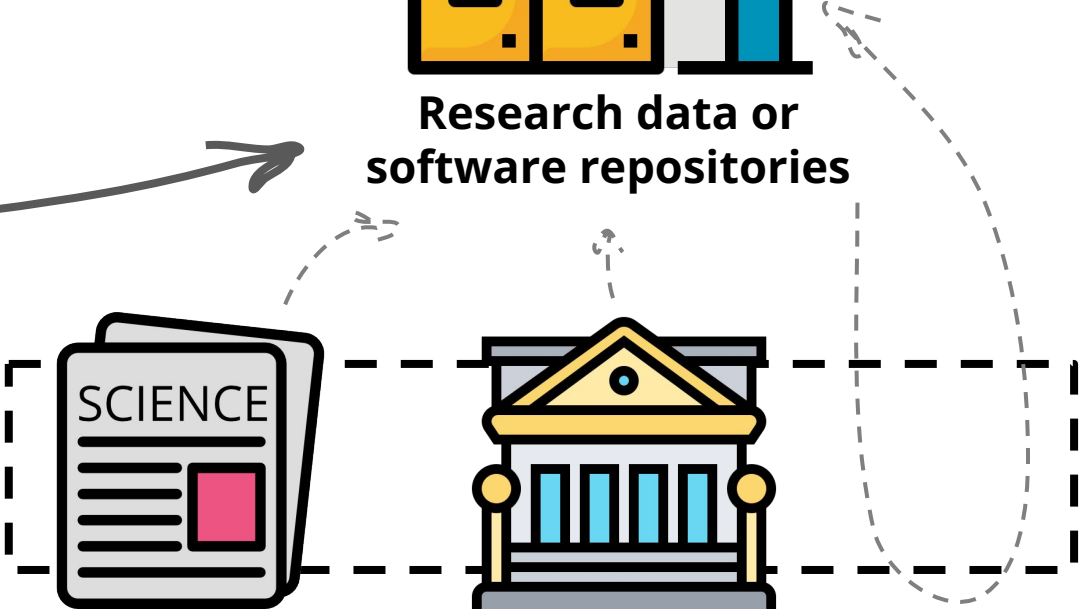
Reviewers



Journals



Institutes

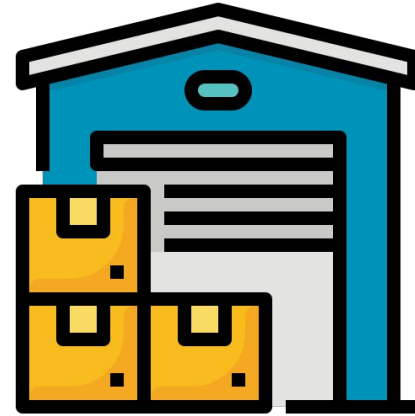
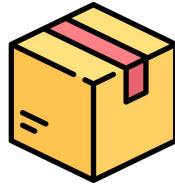


Presentation agenda



Researchers

Research data
and code



Research data or
software repositories

What can
repositories do to
facilitate the code
review process?

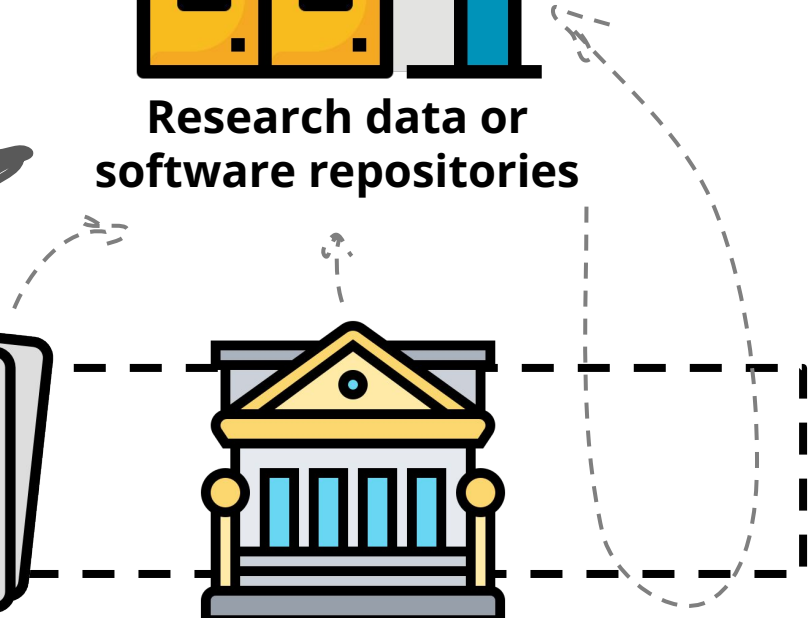
Reviewers



Journals



Institutes



Research code on Harvard Dataverse

```
### TABLE 4: Proportion of MEPs Giving a Speech as a Function of
### Voting and Candidate Selection Mechanisms
### (Vote-speech sample from 6th EP, 2004-2005)
```

```
table(natrebel,centralizedcandidateselection,gavespeech)
```

```
### TABLE 4: two-sample test of proportions
```

```
t.test(gavespeech[natrebel==1 & centralizedcandidateselection==0],
gavespeech[natrebel==0 & centralizedcandidateselection==0],alternative="two.sided",
var.equal=TRUE,conf.level=0.95)
```

```
t.test(gavespeech[natrebel==1 & centralizedcandidateselection==1],
gavespeech[natrebel==0 & centralizedcandidateselection==1],alternative="two.sided",
var.equal=TRUE,conf.level=0.95)
```

```
### WEBAPPENDIX Table 1: Relationship between Voting Decisions
### and Legislative Speeches (Vote-speech sample from 6th EP, 2004-2005)
```

```
table(natrebel,gavespeech)
```

```
### WEBAPPENDIX Table 1: two-sample test of proportions
```

```
t.test(gavespeech[natrebel==0],gavespeech[natrebel==1],
alternative="two.sided",var.equal=TRUE,conf.level=0.95)
```

```
detach(ep6data)
```

```
#####
### AGGREGATE ANALYSIS EP5 #####
#####
```

```
attach(ep5data)
```

```
##### TABLE 5: Negative Binomial MODEL A
```

```
model.A<-zelig(epwebsitespeechcount~ pernpepg+ perepgnp+ tenure+absent+rapporateurshipcount+ partyleader +epleader+ numcommem+ numcommo
ader+groupsize+ natpartyperc +candidateselection+ pernpepg:candidateselection,model="negbin",data= ep5data ,save.data = TRUE)
summary(model.A)
```



```
#Generate new indicators and fill in values
```

```
data$gate<-NA
data$building<-NA
data$split<-NA
data$multi<-NA
data$country_1<-NA
data$country_2<-NA
```

```
data$borderid[data$odd==1]<-paste(data$borderid[data$odd==1],"_1",sep="")
data$gate[data$odd==1]<-data$gate_1[data$odd==1]+1
data$building[data$odd==1]<-data$numbld_1[data$odd==1]+1
data$split[data$odd==1]<-data$split_1[data$odd==1]+1
data$multi[data$odd==1]<-data$multilane_1[data$odd==1]+1
data$country_1[data$odd==1]<-data$country1[data$odd==1]
data$country_2[data$odd==1]<-data$country2[data$odd==1]
```

```
data$borderid[data$odd==0]<-paste(data$borderid[data$odd==0],"_2",sep="")
data$gate[data$odd==0]<-data$gate_2[data$odd==0]+1
data$building[data$odd==0]<-data$numbld_2[data$odd==0]+1
data$split[data$odd==0]<-data$split_2[data$odd==0]+1
data$multi[data$odd==0]<-data$multilane_2[data$odd==0]+1
data$country_1[data$odd==0]<-data$country2[data$odd==0]
data$country_2[data$odd==0]<-data$country1[data$odd==0]
```

```
data$country1<-data$country_1
data$country2<-data$country_2
data$country_1<-NULL
data$country_2<-NULL
```

```
####
```

```
#1.2: Interpolating missing values
```

```
####
```

```
data<-arrange(data,borderid,coder,assignment,year)
```

```
for(jj in 41:44){
```

```
  for(ii in 2:nrow(data)){
```

```
    data[ii,jj]<-ifelse(is.na(data[ii,jj]) & !is.na(data[ii-1,jj]) &
      data$borderid[ii]==data$borderid[ii-1] &
      data$coder[ii]==data$coder[ii-1] &
      data$assignment[ii]==data$assignment[ii-1],
      data[ii-1,jj],data[ii,jj])
```

```
  }
```

```
data<-arrange(data,borderid,coder,assignment,-year)
```

```
for(jj in 41:44){
```

```
  for(ii in 2:nrow(data)){
```

```
    data[ii,jj]<-ifelse(is.na(data[ii,jj]) & !is.na(data[ii-1,jj]) &
      data$borderid[ii]==data$borderid[ii-1] &
      data$coder[ii]==data$coder[ii-1] &
      data$assignment[ii]==data$assignment[ii-1],
      data[ii-1,jj],data[ii,jj])
```

Research code on Harvard Dataverse

TABLE 4: Proportion of MEPs Giving a Speech as a Function of

#Generate new indicators and fill in values

```
#effect of isp (through proportion renewable, electricity per gdp [itself direct and through price, and direct effect)
no.isp.price<-mod1$coef[1]+mod1$coef[2]*mean(POLICYII$rpsprop)+mod1$coef[3]*0+mod1$coef[4]*mean(POLICYII$ipp)+mod1$coef[5]*0+mod1$coef[6]*mean(POLICYII$gasPrice);no.isp.price
no.isp.renew<-pnorm(mod2$coef[1]+mod2$coef[2]*mean(POLICYII$rpsprop)+mod2$coef[3]*0+mod2$coef[4]*mean(POLICYII$ipp)+mod2$coef[5]*0+mod2$coef[6]*mean(POLICYII$gdp)+mod2$coef[7]*mean(POLICYII$gasPrice));no.isp.renew
no.isp.eGDP<-mod3$coef[1]+mod3$coef[2]*0+mod3$coef[3]*mean(POLICYII$ipp)+mod3$coef[4]*0+mod3$coef[5]*no.isp.price+mod3$coef[6]*mean(POLICYII$gasPrice);no.isp.eGDP
no.isp<-exp(mod4$coef[1]+mod4$coef[2]*mean(POLICYII$rpsprop)+mod4$coef[3]*0+mod4$coef[4]*mean(POLICYII$ipp)+mod4$coef[5]*0+mod4$coef[6]*mean(POLICYII$gdp)+mod4$coef[7]*no.isp.renew+mod4$coef[8]*no.isp.eGDP+mod4$coef[9]*mean(POLICYII$gasPrice))

yes.isp.price<- mod1$coef[1]+mod1$coef[2]*mean(POLICYII$rpsprop)+mod1$coef[3]*0+mod1$coef[4]*mean(POLICYII$ipp)+mod1$coef[5]*1+mod1$coef[6]*mean(POLICYII$gasPrice);yes.isp.price
yes.isp.renew<-pnorm(mod2$coef[1]+mod2$coef[2]*mean(POLICYII$rpsprop)+mod2$coef[3]*0+mod2$coef[4]*mean(POLICYII$ipp)+mod2$coef[5]*1+mod2$coef[6]*mean(POLICYII$gdp)+mod2$coef[7]*mean(POLICYII$gasPrice));yes.isp.renew
yes.isp.eGDP<- mod3$coef[1]+mod3$coef[2]*0+mod3$coef[3]*mean(POLICYII$ipp)+mod3$coef[4]*0+mod3$coef[5]*yes.isp.price+mod3$coef[6]*mean(POLICYII$gasPrice);yes.isp.eGDP
yes.isp<-exp(mod4$coef[1]+mod4$coef[2]*mean(POLICYII$rpsprop)+mod4$coef[3]*0+mod4$coef[4]*mean(POLICYII$ipp)+mod4$coef[5]*1+mod4$coef[6]*mean(POLICYII$gdp)+mod4$coef[7]*yes.isp.renew+mod4$coef[8]*yes.isp.eGDP+mod4$coef[9]*mean(POLICYII$gasPrice))

no.isp.yes.isp
100*(no.isp-yes.isp)/no.isp
```

```
gavespeech[nat rebel==0 & realizedcandidateselection==1, alternative="two.sided",
var.equal=TRUE, conf.level=0.95]
```

WEBAPPENDIX Table 1: Relationship between Voting Decisions
and Legislative Speeches (Vote-speech sample from 6th EP, 2004–2005)

```
table(nat rebel, gavespeech)
```

WEBAPPENDIX Table 1: two-sample test of p

```
t.test(gavespeech[nat rebel==0], gavespeech[nat rebel==1],
alternative="two.sided", var.equal=TRUE)
```

```
detach(ep6data)
```

```
#####
#### AGGREGATE ANALYSIS EP5 #####
#####
```

```
attach(ep5data)
```

```
##### TABLE 5: Negative Binomial MODEL A
model.A<-zelig(epwebsitespeechcount~ pernpegg+
ader+groupsize+ natpartyperc +candidateselection
summary(model.A)
```



```
data$country_2[data$odd==1]<-data$country2[data$odd==1]
```

```
data$borderid[data$odd==0]<-paste(data$borderid[data$odd==0], "_2", sep="")
data$gate[data$odd==0]<-data$gate_2[data$odd==0]+1
data$building[data$odd==0]<-data$numld_2[data$odd==0]+1
data$split[data$odd==0]<-data$split_2[data$odd==0]+1
data$multi[data$odd==0]<-data$multilane_2[data$odd==0]+1
data$country_1[data$odd==0]<-data$country2[data$odd==0]
data$country_2[data$odd==0]<-data$country1[data$odd==0]
```

```
1 se.est.plot <- c(sd(brazil.data$voteintent[brazil.data$education < 4 & brazil.data$cred_vs_less==0], na.rm=T)/
2 sqrt(sum(is.na(brazil.data$voteintent[brazil.data$education < 4 & brazil.data$cred_vs_less==0])=0)),
3 sd(brazil.data$voteintent[brazil.data$education < 4 & brazil.data$cred_vs_less==1], na.rm=T)/
4 sqrt(sum(is.na(brazil.data$voteintent[brazil.data$education < 4 & brazil.data$cred_vs_less==1])=0)),
5 sd(brazil.data$voteintent[brazil.data$education==4 & brazil.data$cred_vs_less==0], na.rm=T)/
6 sqrt(sum(is.na(brazil.data$voteintent[brazil.data$education==4 & brazil.data$cred_vs_less==0])=0)),
7 sd(brazil.data$voteintent[brazil.data$education==4 & brazil.data$cred_vs_less==1], na.rm=T)/
8 sqrt(sum(is.na(brazil.data$voteintent[brazil.data$education==4 & brazil.data$cred_vs_less==1])=0)),
9 sd(argentina.data$voteintent[argentina.data$education < 3 & argentina.data$cred_vs_less==0], na.rm=T)/
10 sqrt(sum(is.na(argentina.data$voteintent[argentina.data$education < 3 & argentina.data$cred_vs_less==0])=0)),
11 sd(argentina.data$voteintent[argentina.data$education < 3 & argentina.data$cred_vs_less==1], na.rm=T)/
12 sqrt(sum(is.na(argentina.data$voteintent[argentina.data$education < 3 & argentina.data$cred_vs_less==1])=0)),
13 sd(argentina.data$voteintent[argentina.data$education==3 & argentina.data$cred_vs_less==0], na.rm=T)/
14 sqrt(sum(is.na(argentina.data$voteintent[argentina.data$education==3 & argentina.data$cred_vs_less==0])=0)),
15 sd(argentina.data$voteintent[argentina.data$education==3 & argentina.data$cred_vs_less==1], na.rm=T)/
16 sqrt(sum(is.na(argentina.data$voteintent[argentina.data$education==3 & argentina.data$cred_vs_less==1])=0))
```

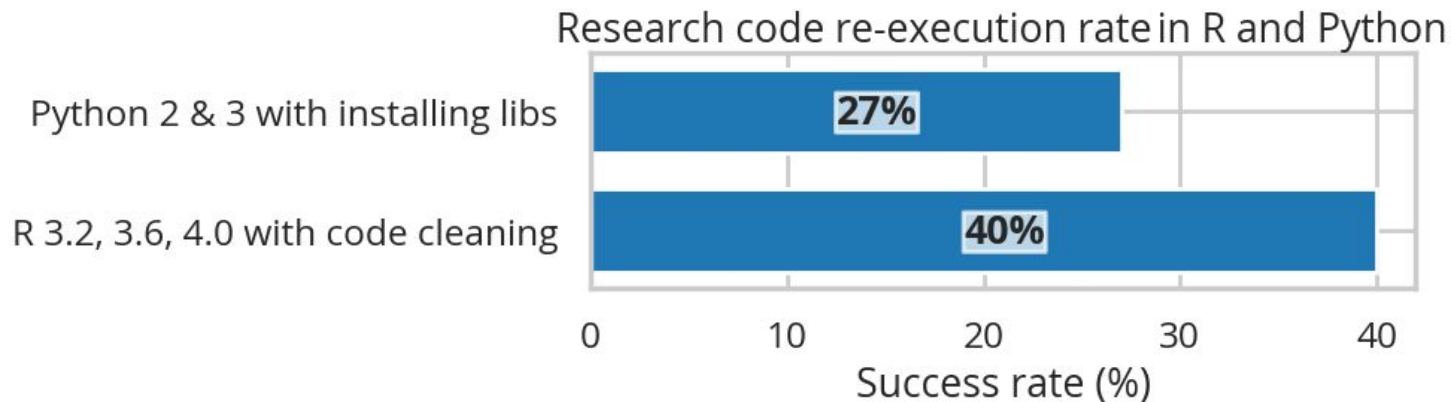
```
data[ii-1, jj] &
rid[ii-1] &
-1] &
ignment[ii-1,
```

```
data[ii-1, jj] &
rid[ii-1] &
-1] &
```

```
data[ii-1, jj], data[ii, jj]
```

Research code re-execution

Most code files fail when re-executed out-of-the-box, even with the pre-installation of used libraries [1,2].

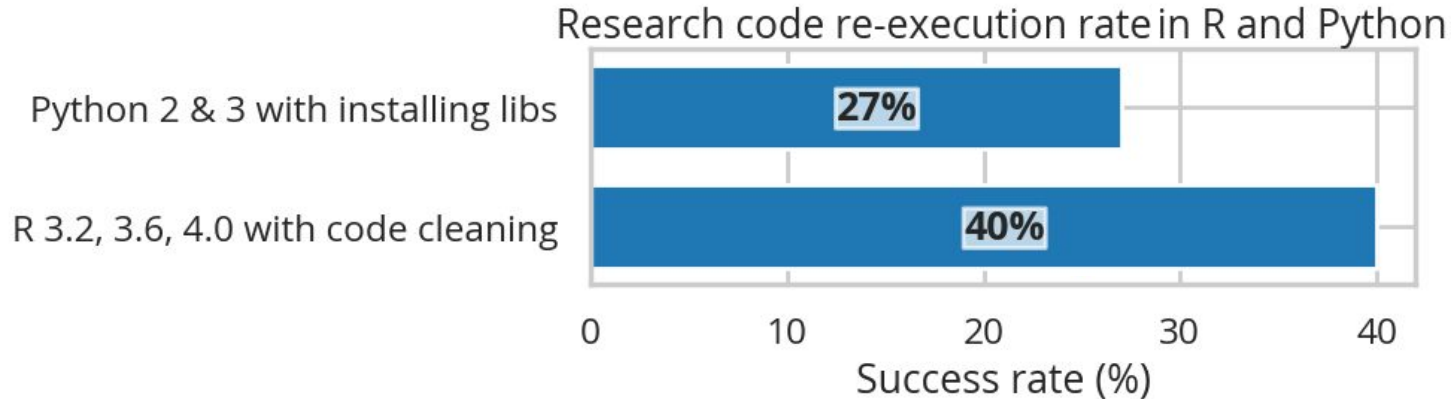


[1] Trisovic, Ana, et al. "Repository Approaches to Improving Quality of Shared Data and Code." Data 6.2 (2021): 15.

[2] Trisovic, Ana, et al. "A large-scale study on research code quality and execution." arXiv preprint arXiv:2103.12793 (2021).

Research code re-execution

Most code files fail when re-executed out-of-the-box, even with the pre-installation of used libraries [1,2].



It is hard to re-execute “old” code, but many common errors can be avoided!

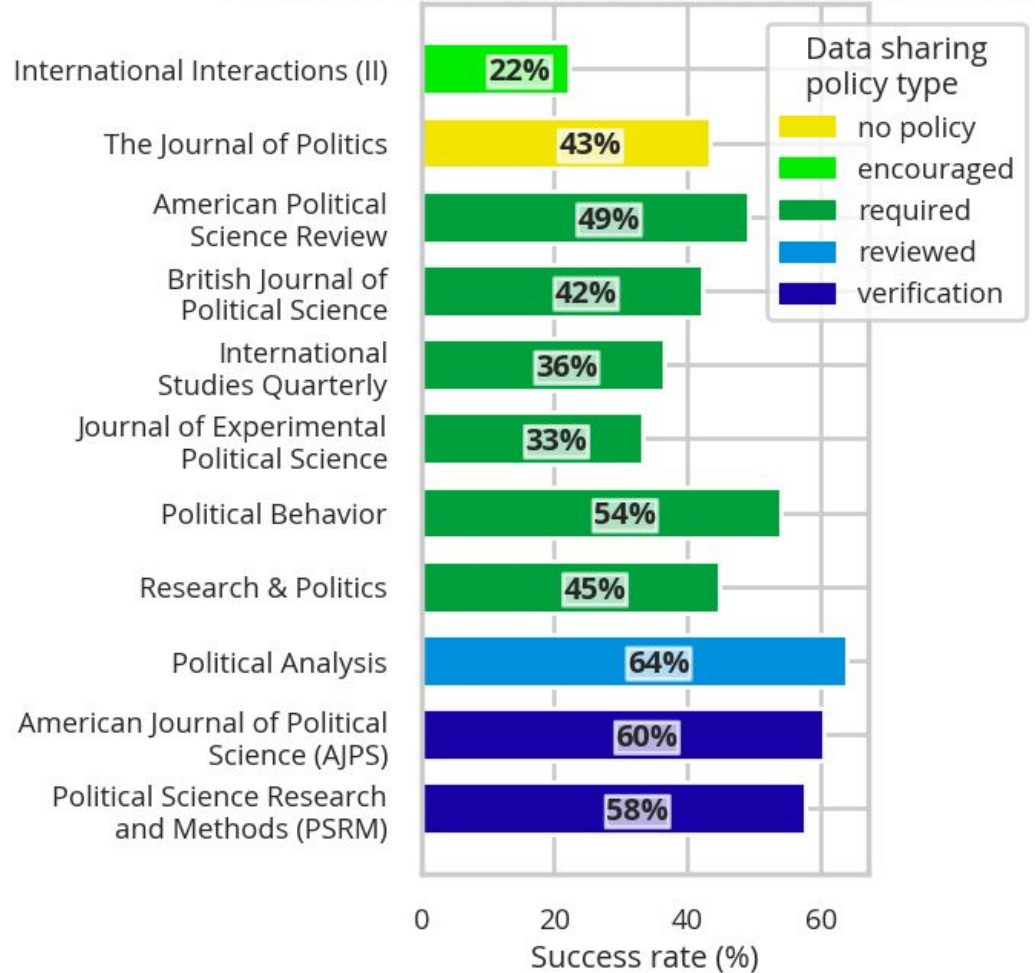
[1] Trisovic, Ana, et al. "Repository Approaches to Improving Quality of Shared Data and Code." Data 6.2 (2021): 15.

[2] Trisovic, Ana, et al. "A large-scale study on research code quality and execution." arXiv preprint arXiv:2103.12793 (2021).

Journals with stricter data policies have higher rate of executable code

Had some code review!

Portion of replication datasets with re-executable code files



**Can we learn from
industry code
review?**

Research code vs. industry code

Industry development teams use advanced tools to facilitate code review such as continuous integration and containers

In most cases, students and early-career researchers write research code (sometimes completely new to programming)

Research code vs. industry code

Industry development teams use advanced tools to facilitate code review such as continuous integration and containers

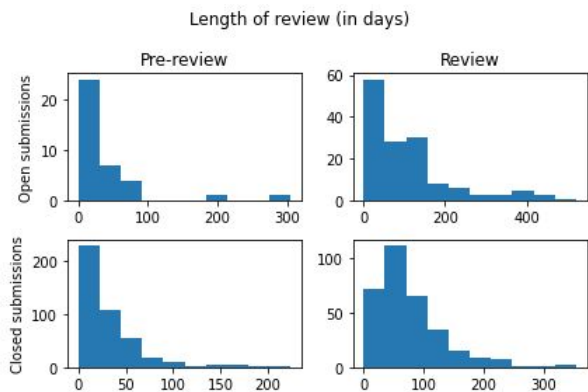
In most cases, students and early-career researchers write research code (sometimes completely new to programming)

Volunteers review research code (seeing it for the first time), while development team members (who are already familiar with software) review new (small) code contributions

Lifetime of research code is less than industry code so there are less incentives to keep it clean

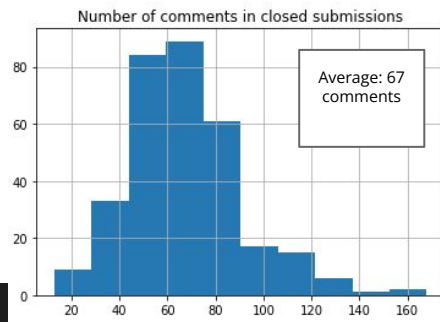
Publishing research software with the Journal of Open Source Software (JOSS)

- Software as a recognized output of research
- Retrieved 1000 closed and 182 open issues (paper submission and review) containing 25,382 comments.



Mean review time $30.5 + 77.4 = 107.9$ days
Median review time $21.0 + 63.0 = 84.0$ days

Assigning reviewers and the review is happening in GitHub issue comments



The screenshot shows a GitHub issue thread with several comments. The issue is titled '@whedon re-invite @linuxscout as reviewer'. The thread includes comments from whedon, linuxscout, ChrisW09, and pps121. A red arrow points from the text 'Assigning reviewers and the review is happening in GitHub issue comments' to the comment where whedon assigned linuxscout as a reviewer on Sep 17.

arfon commented on Sep 17 Member

@whedon re-invite @linuxscout as reviewer

whedon commented on Sep 17 Collaborator Author

OK, the reviewer has been re-invited.

linuxscout commented on Sep 17

ok, thanks.

linuxscout commented on Sep 17

Hi,
I finished the review.

ChrisW09 commented on Sep 17

@linuxscout thanks a lot for being that quick with the review. We really appreciated your comments and the fast review process!

ChrisW09 commented on Sep 17

@pps121 we are already looking forward to your comments. Please let us know if anything is unclear :) thanks!

whedon assigned linuxscout on Sep 17

ChrisW09 commented 22 days ago

Hi, @pps121 how are things going with the review? Please let us know if anything is unclear. thanks!

arfon commented 16 days ago Member

FYI I just emailed @pps121 to see when they might be able to complete their review by.

ChrisW09 commented 16 days ago

Thank you @arfon! We are looking forward to your feedback @pps121.

arfon commented 16 days ago Member

I just heard back from @pps121 and they are committed to completing their review soon, but are currently busy with school/university commitments.

ChrisW09 commented 16 days ago

Great, thank you both @arfon and @pps121!

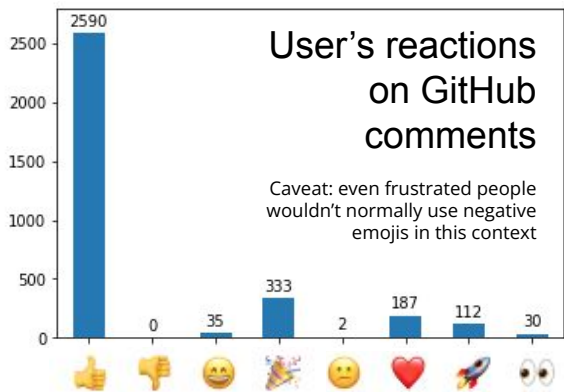
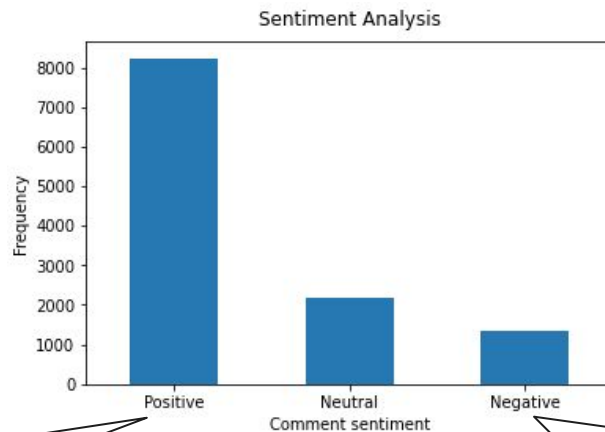
pps121 commented 10 days ago

When I ran the code: from nltk.corpus import reuters
it gave me LookupError as below :

Resource reuters not found.
Please use the NLTK Downloader to obtain the resource:

How reviewers feel about reviewing software?

- Basic NLP analysis of the content of JOSS issue comments in completed submissions.
 - Comments by the bot @whedon were excluded.
- The sentiment seems mostly positive!



xuanxu: Great! Thank you both!
bradkav: I'm also now happy to recommend `sntools` for publication. Well done!
neuromusic: Changes look great!
Yurlungur: Perfect!

danielskatz: This may have just been very bad timing
SteveMacenski: Hi, just wanted to touch base on this - any progress?
simonom: I have completed my review, but my invitation to review has expired so I can't check off the checklist :(
sgrieve: I'm afraid I don't have any capacity this month. Apologies!



Word cloud of JOSS submission comments (reviews)

Ideas to make code review easy at research repositories

1) Checklist for code reviewers / data curators

Review checklist

✨ Important: Please do not use the *Convert to issue* functionality when working through this checklist, instead, please open any new issues associated with your review [in the software repository associated with the submission](#). ✨

Conflict of interest

- I confirm that I have read the [JOSS conflict of interest \(COI\) policy](#) and that: I have no COIs with reviewing this work or that any perceived COIs have been waived by JOSS for the purpose of this review.

Code of Conduct

- I confirm that I read and will adhere to the [JOSS code of conduct](#).

General checks

- Repository:** Is the source code for this software available at the [repository url](#)?
- License:** Does the repository contain a plain-text LICENSE file with the contents of an [OSI approved](#) software license?
- Contribution and authorship:** Has the submitting author (@[hcdenbakker](#)) made major contributions to the software? Does the full list of paper authors seem appropriate and complete?
- Substantial scholarly effort:** Does this submission meet the scope eligibility described in the [JOSS guidelines](#)

Functionality

- Installation:** Does installation proceed as outlined in the documentation?
- Functionality:** Have the functional claims of the software been confirmed?
- Performance:** If there are any performance claims of the software, have they been

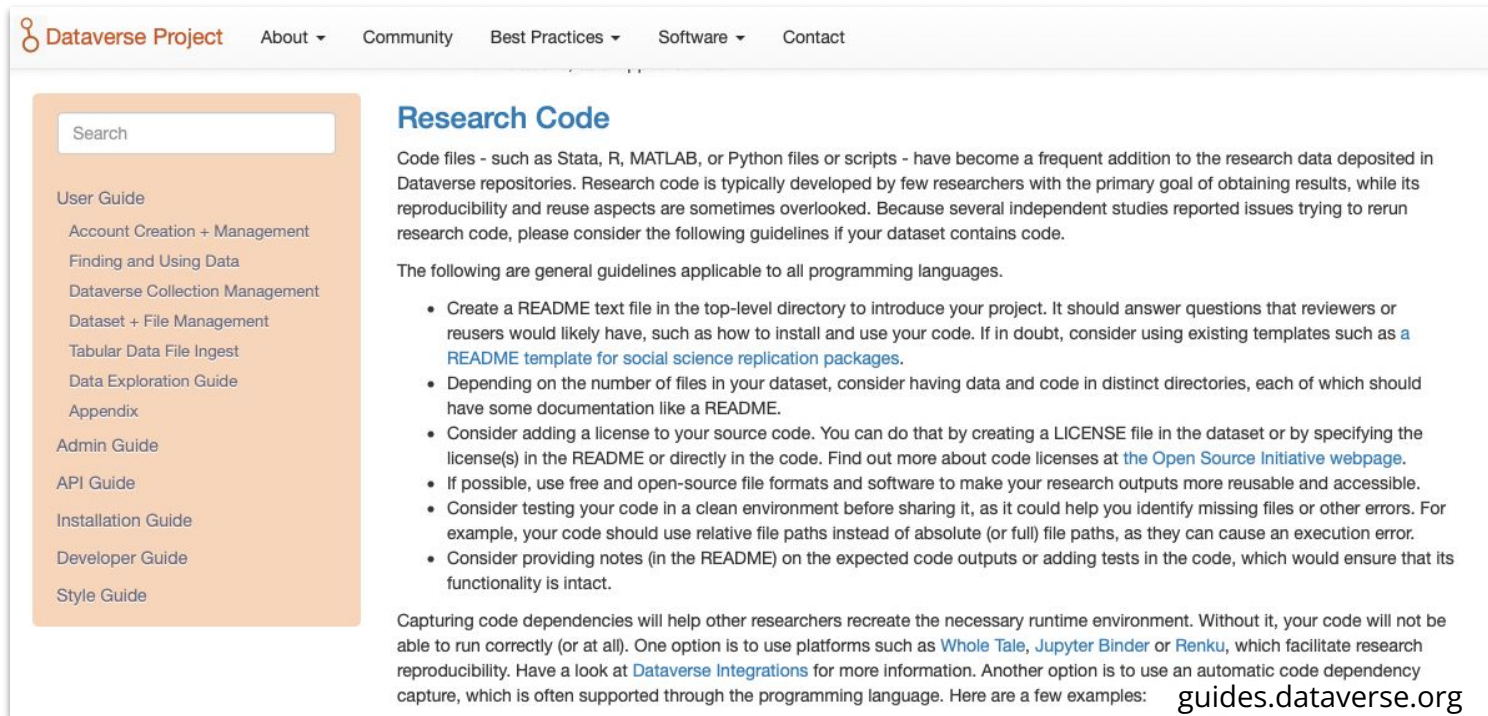
Documentation

- A statement of need:** Do the authors clearly state what problems the software is designed to solve and who the target audience is?
- Installation instructions:** Is there a clearly-stated list of dependencies? Ideally these should be handled with an automated package management solution.
- Example usage:** Do the authors include examples of how to use the software (ideally to solve real-world analysis problems).
- Functionality documentation:** Is the core functionality of the software documented to a satisfactory level (e.g., API method documentation)?
- Automated tests:** Are there automated tests or manual steps described so that the functionality of the software can be verified?
- Community guidelines:** Are there clear guidelines for third parties wishing to 1) Contribute to the software 2) Report issues or problems with the software 3) Seek support

Software paper

- Summary:** Has a clear description of the high-level functionality and purpose of the software for a diverse, non-specialist audience been provided?
- A statement of need:** Does the paper have a section titled 'Statement of Need' that clearly states what problems the software is designed to solve and who the target audience is?
- State of the field:** Do the authors describe how this software compares to other commonly-used packages?
- Quality of writing:** Is the paper well written (i.e., it does not require editing for structure, language, or writing quality)?
- References:** Is the list of references complete, and is everything cited appropriately that should be cited (e.g., papers, datasets, software)? Do references in the text use the proper [citation syntax](#)?

2) Guidelines for code depositors



The screenshot shows the Dataverse Project website. The navigation bar includes the logo, 'Dataverse Project', and links for 'About', 'Community', 'Best Practices', 'Software', and 'Contact'. A search bar is located in the top left. A sidebar on the left contains a list of guides: 'User Guide', 'Account Creation + Management', 'Finding and Using Data', 'Dataverse Collection Management', 'Dataset + File Management', 'Tabular Data File Ingest', 'Data Exploration Guide', 'Appendix', 'Admin Guide', 'API Guide', 'Installation Guide', 'Developer Guide', and 'Style Guide'. The main content area is titled 'Research Code' and contains the following text and list:

Code files - such as Stata, R, MATLAB, or Python files or scripts - have become a frequent addition to the research data deposited in Dataverse repositories. Research code is typically developed by few researchers with the primary goal of obtaining results, while its reproducibility and reuse aspects are sometimes overlooked. Because several independent studies reported issues trying to rerun research code, please consider the following guidelines if your dataset contains code.

The following are general guidelines applicable to all programming languages.

- Create a README text file in the top-level directory to introduce your project. It should answer questions that reviewers or reusers would likely have, such as how to install and use your code. If in doubt, consider using existing templates such as a [README template for social science replication packages](#).
- Depending on the number of files in your dataset, consider having data and code in distinct directories, each of which should have some documentation like a README.
- Consider adding a license to your source code. You can do that by creating a LICENSE file in the dataset or by specifying the license(s) in the README or directly in the code. Find out more about code licenses at [the Open Source Initiative webpage](#).
- If possible, use free and open-source file formats and software to make your research outputs more reusable and accessible.
- Consider testing your code in a clean environment before sharing it, as it could help you identify missing files or other errors. For example, your code should use relative file paths instead of absolute (or full) file paths, as they can cause an execution error.
- Consider providing notes (in the README) on the expected code outputs or adding tests in the code, which would ensure that its functionality is intact.

Capturing code dependencies will help other researchers recreate the necessary runtime environment. Without it, your code will not be able to run correctly (or at all). One option is to use platforms such as [Whole Tale](#), [Jupyter Binder](#) or [Renku](#), which facilitate research reproducibility. Have a look at [Dataverse Integrations](#) for more information. Another option is to use an automatic code dependency capture, which is often supported through the programming language. Here are a few examples: guides.dataverse.org

3) Automatic code style assessment

- Automatic code style assessment informs the depositors of the readability of their code.
- There are existing code formatting tools that can be encouraged or recommended (`lintR`, `pycodestyle`, even in-browser tools).

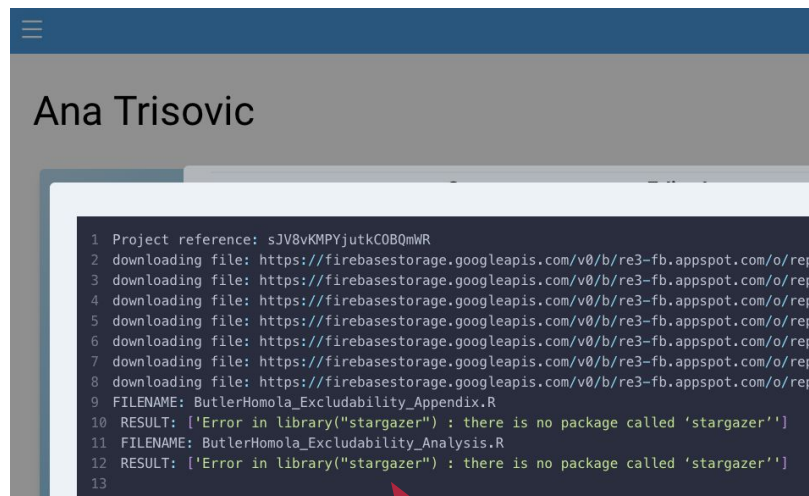
The screenshot shows a web interface for code readability assessment. At the top right, there are links for "My Account" and "Sign out". Below these is a "Get Code Rating" button. A progress bar shows a score of 6.34. A "Consider:" box contains three bullet points: "breaking up your lines", "removing any irrelevant parentheses in your lines", and "diminishing the periods in your lines". A red arrow points from this box to the code in the editor. At the bottom, there is an "Upload File" button.

```
15
16 ## read in "Names" dataset (Table 1 & Figure 1)
17 data <- read.csv("ButlerHomola_Excludability_Names.csv")
18
19 ## Remove name with no match (Alaliyah)
20 data <- data[-2,]
21
22 ## Create political resources factor score
23 resources <- prcomp(~ Income + Education + Housing1 + Turnout,
24                   data=data, na.action = na.exclude)
25 data$resources <- -1*(resources$x[,1])
26 ## Eigenvalue of first factor
27 (resources$sdev^2*4/sum(resources$sdev^2))[1]
28 ## 2.73
29
30 ## check that high factor values equal higher values on the component part
31 cor(data$Income, data$resources, use="complete.obs")
32 cor(data$Education, data$resources, use="complete.obs")
33 cor(data$Housing1, data$resources, use="complete.obs")
34 cor(data$Turnout, data$resources, use="complete.obs")
35
36 ## correlation matrix
37 cor(cbind(data$resources, data$Income, data$Education, data$Housing1,
38          data$Turnout), use="complete.obs", method="spearman")
39
```

Code readability
test at upload

4) Automatic re-execution (for free software and small-scale studies)

- Enabling an automatic code re-execution test could fast identify missing files and other common errors.
- It could help with the documentation of analysis flow as the researchers would need to add re-execution commands.
- Code would run out-of-the-box for reviewers and future reusers.



```
1 Project reference: sJV8vKMPYjutrCOBQmWR
2 downloading file: https://firebasestorage.googleapis.com/v0/b/re3-fb.appspot.com/o/rep
3 downloading file: https://firebasestorage.googleapis.com/v0/b/re3-fb.appspot.com/o/rep
4 downloading file: https://firebasestorage.googleapis.com/v0/b/re3-fb.appspot.com/o/rep
5 downloading file: https://firebasestorage.googleapis.com/v0/b/re3-fb.appspot.com/o/rep
6 downloading file: https://firebasestorage.googleapis.com/v0/b/re3-fb.appspot.com/o/rep
7 downloading file: https://firebasestorage.googleapis.com/v0/b/re3-fb.appspot.com/o/rep
8 downloading file: https://firebasestorage.googleapis.com/v0/b/re3-fb.appspot.com/o/rep
9 FILENAME: ButlerHomola_Excludability_Appendix.R
10 RESULT: ['Error in library("stargazer") : there is no package called 'stargazer'']
11 FILENAME: ButlerHomola_Excludability_Analysis.R
12 RESULT: ['Error in library("stargazer") : there is no package called 'stargazer'']
13
```

Code re-execution
test at upload

5*) “Walk me through your code” video

- Media files as part of publication (video summaries of articles (i.e., HDSR))
- Creating a video presentation of code could take a few hours for its creator, but probably save twice as much time for each code reviewer and reuser
- Creating video presentations is easy with Zoom!

HDSR Search

HOME JOURNAL CATEGORIES TOPICS MEDIA FEATURES PODCAST SUI

Issue 2.4, Fall 2020 Published on Dec 21, 2020 DOI 10.1162/99608f92.250f99

Reproducibility and Replication of Experimental Particle Physics Results

by Thomas R. Junk and Louis Lyons

Published on Dec 21, 2020

last released 9 months ago

ABSTRACT

Recently, much attention has been focused on the replicability of scientific results, causing scientists, statisticians, and journal editors to examine closely their methodologies and publishing criteria. Experimental particle physicists have been aware of the precursors of nonreplicable research for many decades and have many safeguards to ensure that the published results are as reliable as possible. The experiments require large investments of time and effort to design, construct, and operate. Large collaborations produce and check the results, and many papers are signed by more than 3,000 authors. This article gives an introduction to what experimental particle physics is and to some of the tools that are used to analyze the data. It describes the procedures used to ensure that results can be computationally reproduced, both by collaborators and by noncollaborators. It describes the status of publicly available data sets and analysis tools that aid in reproduction and recasting of experimental results. It also describes methods particle physicists use to maximize the reliability of the results, which increases the probability that they can be replicated by other collaborations or even the same collaborations with more data and new personnel. Examples of results that were later found to be false are given, both with failed replication attempts and one with alarmingly successful replications. While some of the characteristics of particle physics experiments are unique, many of the procedures and techniques can be and are used in other fields.

Keywords: reliability, reproducibility, replication, particle physics



Video Abstract



ALFRED P. SLOAN
FOUNDATION



Thank you! Questions?

Email: anatrivic@g.harvard.edu
GitHub & Twitter: [atrisovic](#)